Large websites and tight time limits

On a website with a few thousand media files and a lot of content, a scan is a great deal of work. So that it gets through anyway — including with providers who keep computing time short — Media Organizer does not work in one go, but in **steps**.

What a step is

When you click **Scan**, the dashboard keeps asking the server for a small piece of work: a few hundred files, a few hundred records. After every piece the server writes down how far it got and reports back. How long a scan takes therefore shows in the *number* of steps, not in the length of any one of them. That is the whole point: a server aborts an operation that takes too long, but it does not count how many short ones follow each other.

While the scan runs you can see what it is working on (“Listing files…”, “Collecting references: Articles”), how far the current phase is, what has been found so far, and how long it has been running.

> **The progress is kept in the database, not in the browser.** You can close the window, switch the computer off, or navigate away — the scan then stands still, but it is not lost. The next time you open the dashboard, Media Organizer offers **Resume scan**.

When a step is cut short

If Media Organizer reports that the server cut a step short, the server's time limit has almost always been reached. In that case:

1. Click **Resume scan**. The run carries on from the same place; the work done so far is kept.
2. If it stops again, reduce the step size under **Performance** in the options — see below.
3. If that does not help either, run the scan as a scheduled task or from the command line.

Setting the step size

Options → Performance:

Setting	Default	Meaning
Files per scan step	200	How many files one step records and checks.
Records per scan step	500	How many articles, modules and other records one step searches for references.
Maximum time per scan step	10 seconds	A step ends after this time at the latest, even if the configured amount has not been reached.

Smaller values make every step shorter and the whole run somewhat longer. That is the trade this is about: a scan that takes ten minutes but gets through beats one that stops after three.

The **maximum time** is the safety net for records that turn out to be unexpectedly expensive — an article with a great deal of text, media on network storage. It should sit well below your server's time limit; the default of 10 seconds fits the usual 30.

The second scan is cheaper than the first

Media Organizer computes a checksum for every file — that is how it recognises duplicates and later changes. It is the part that costs the most time, because every file has to be read in full once.

On the next scan that only happens for files whose size or modification date differs. On a website where twenty images were added between two scans, only those twenty are read.

Scanning without a browser

With very large collections the browser is not the best place for a scan. Two ways lead around it:

- **The scheduled task.** It carries a large scan through several runs: each run works for a while and the next one continues. The e-mail arrives when the scan has finished — not

after every run.
- **The command line.** There is no time limit there; the scan runs in one go and shows its progress as it works.

Both share the same progress with the backend. A scan you started in the backend can be finished by the scheduled task — and the other way round.

The log

When something goes wrong technically — a step that was cut short, a search area that could not be read — Media Organizer writes it to the file `jimgo.php` in your site's log folder (the path is under **System → Global Configuration → System → Path to Log Folder**). A line looks like this:

```
2026-09-12T14:22:31+00:00␣ERROR␣A scan step failed. [scan=118 phase=providers error=… where=…]
```

Out of the box only failures are recorded, which in normal operation writes almost nothing. You can change that under **Options → Log** — “Everything” also records the start and end of every scan and is not meant to be left on.

If you report a problem to us, this file is the most useful thing you can send along.

If it still gets stuck

One point for the conversation with your provider: the PHP setting `max_execution_time` is often *not* the cause. It does not count time spent waiting for the database or for file access — and that is most of what a scan does. What usually cuts a scan short is the time limit of the web server in front of it (`fastcgi_read_timeout` in nginx, `request_terminate_timeout` in PHP-FPM). That is what to ask about once the step size no longer helps.

Deutsche Fassung

Revision #3
Created 2026-09-12 14:00:56 UTC by Norbert Graup
Updated 2026-09-12 14:47:05 UTC by Norbert Graup